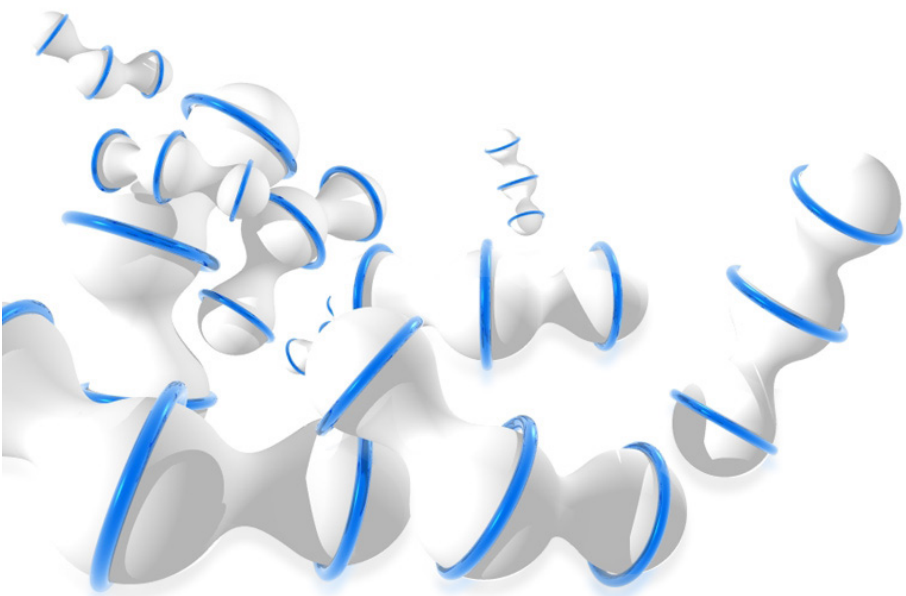


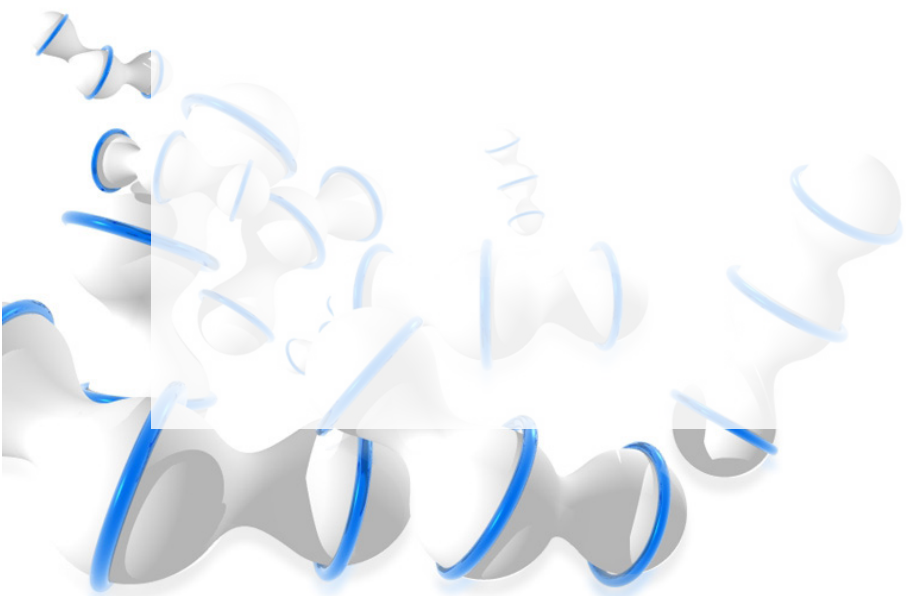
Internationalisierung von Webseiten

„eStudy goes USA“

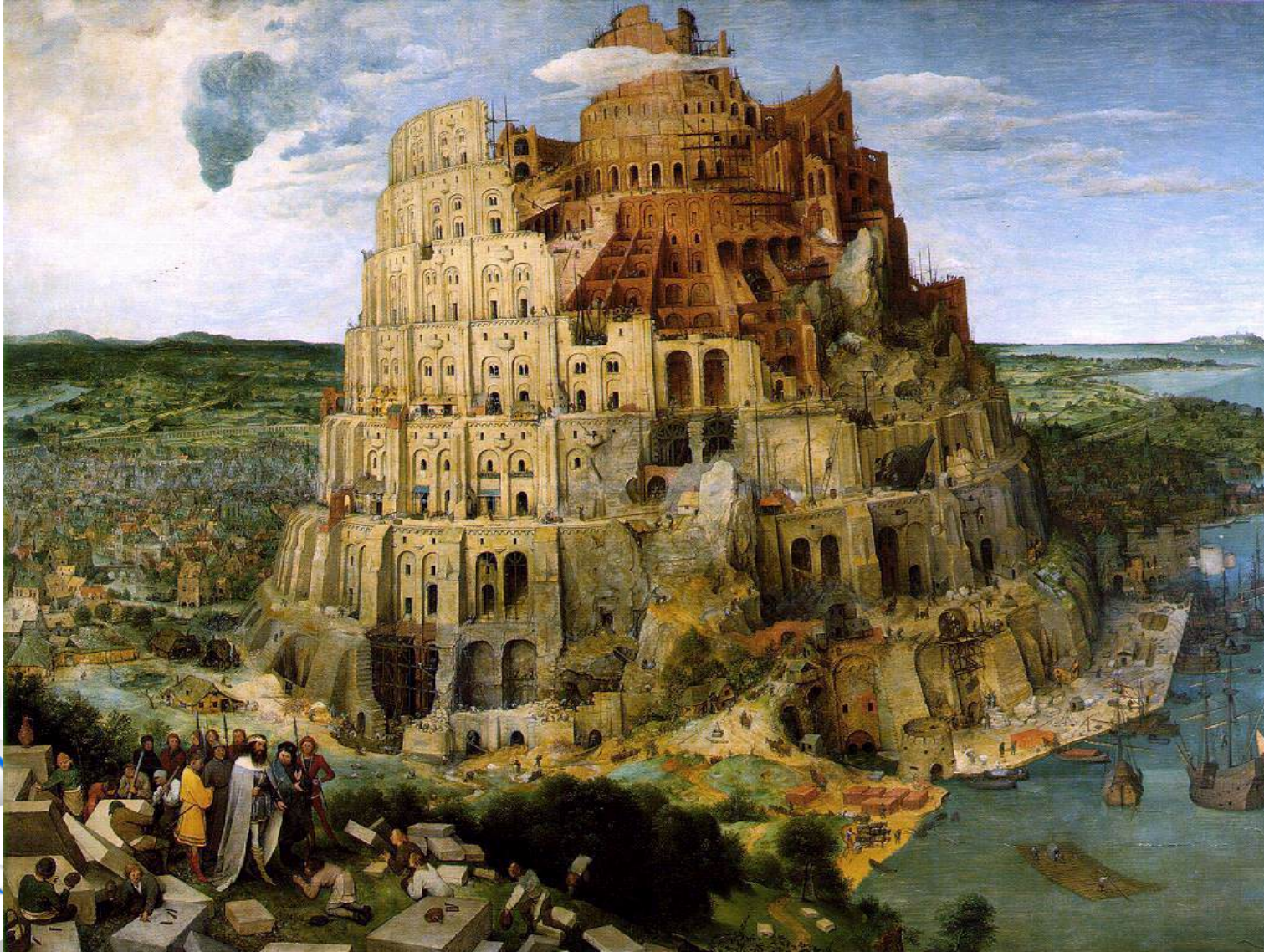


Inhalt

- Einleitung
- Techniken
- Fazit
- Diskussion

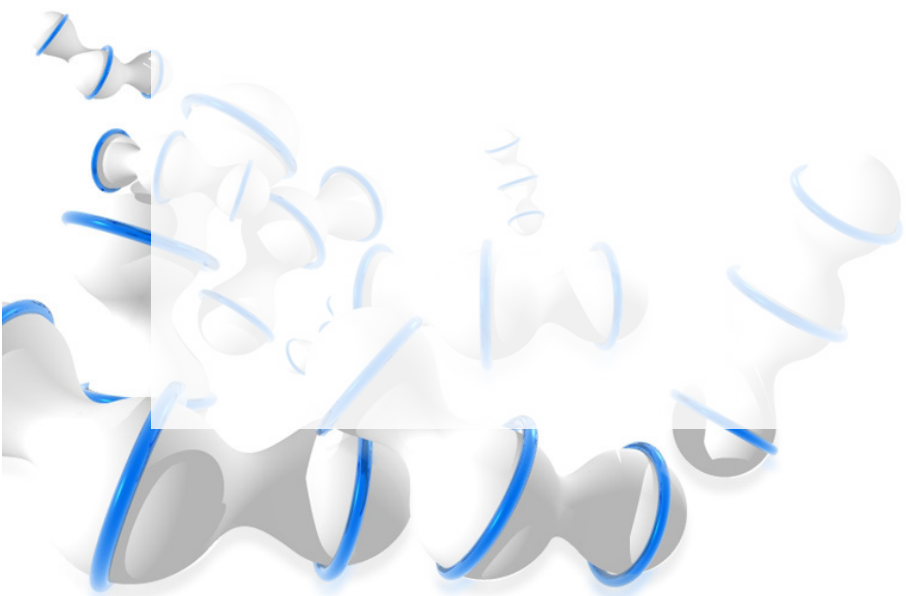


Einleitung



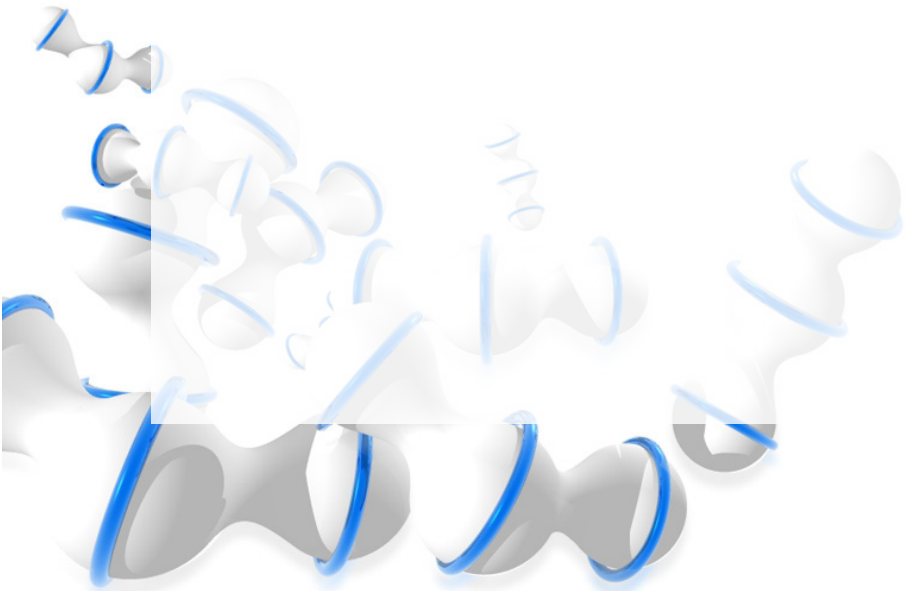
Einleitung

- Sprachen
 - Entstanden durch viele Völker und Kulturen
 - Bilden Barriere für Webentwickler
 - Hohe Wahrscheinlichkeit dass Anwender Inhalt von anderssprachigem Verfasser findet



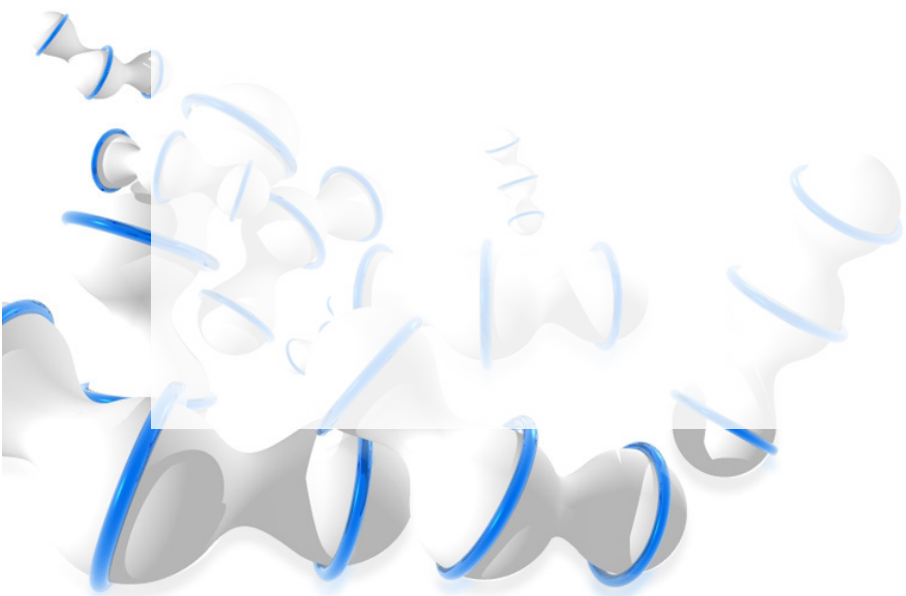
Einleitung

- Internationalisierung?
 - Internet-Sprachgebrauch: *i18n*
 - internationalization
n=18
 - Beschreibt Technik um Software in mehreren Sprachen zugänglich zu machen



Einleitung

- Techniken zur Realisierung
 - Vermeintlich unendlich viele
 - Tatsächlich wenige ähnliche
 - Auswahl -> nach Verwendungszweck



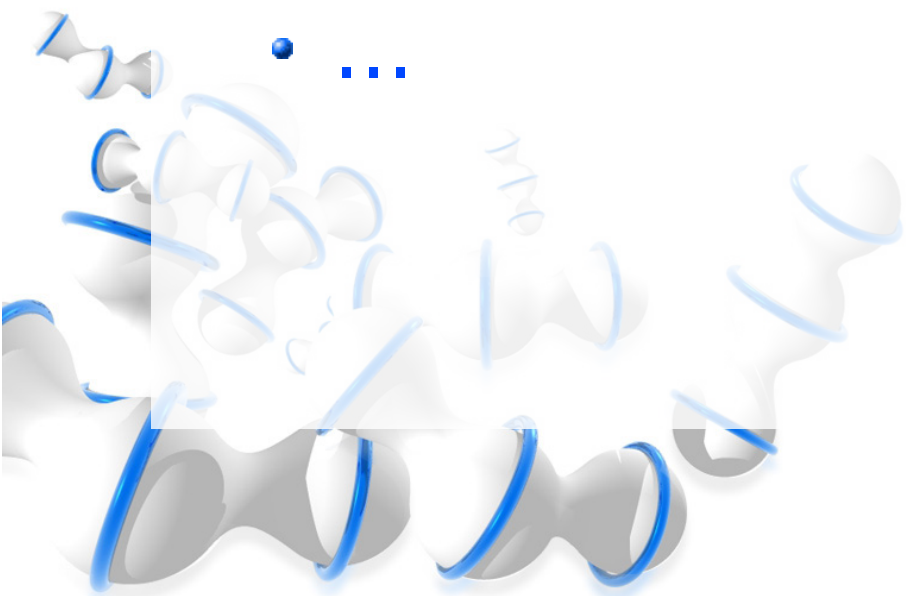
Techniken: Übersicht

- Internationalisierte HTML-Seiten
- Array-basierte Sprachkonzepte
- Gettext
- Stringlisten
- Entities



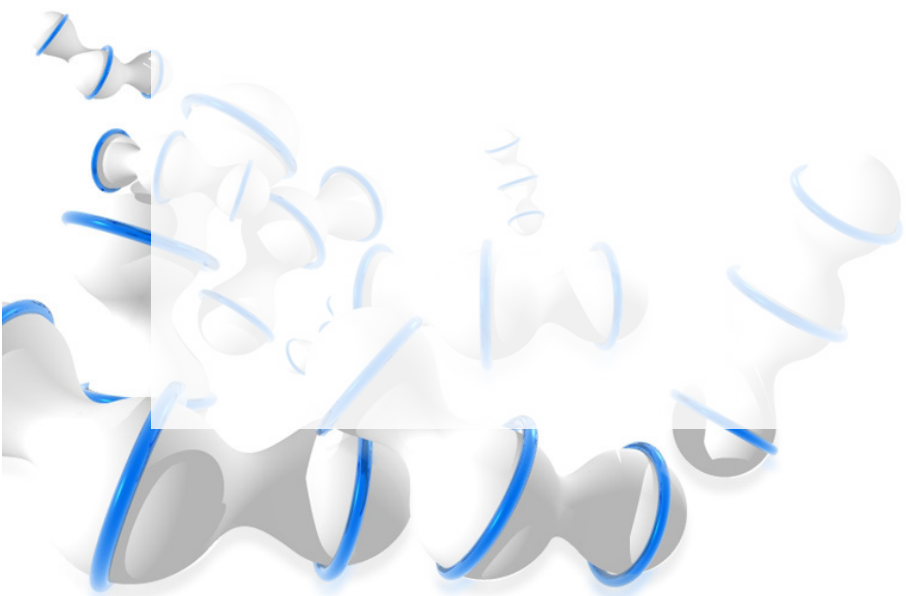
Techniken: Internationalisierte HTML-Seiten

- Erkennung der Sprache durch
 - Aufruf-Parameter
 - Session-Angabe
 - Cookie
 - Browsereinstellungen
 - ...



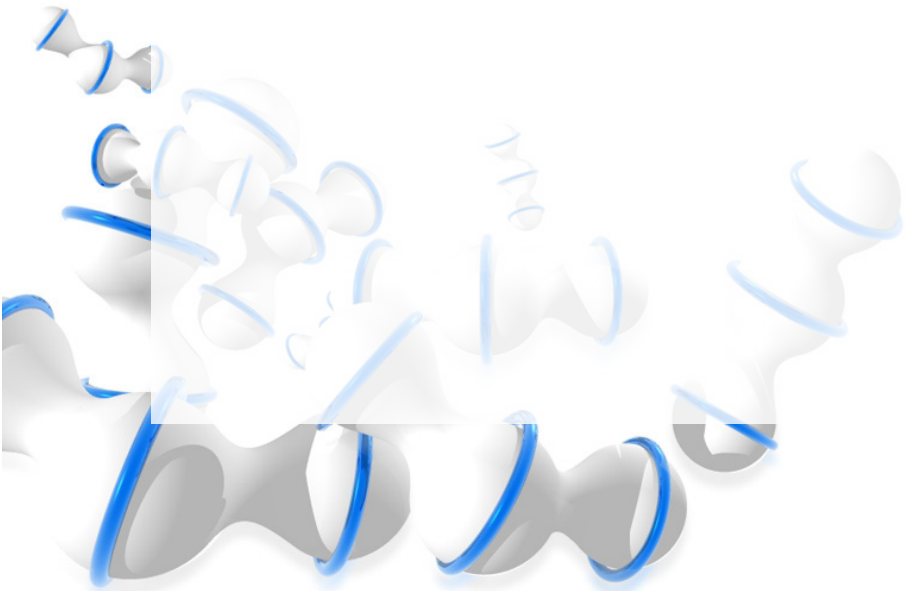
Techniken: Internationalisierte HTML-Seiten

- Auswahl der HTML-Seite
 - In entsprechender Sprache vorhanden -> Anzeige!
 - Nicht vorhanden -> Anzeige in Standardsprache



Techniken: Internationalisierte HTML-Seiten

- Vorteil: einfache Umsetzung
 - Nachteile:
 - dynamische Ausgaben
 - Trennung von Inhalt und Layout
 - Folgen: unwartbare Projektstrukturen!
- } schwer realisierbar!



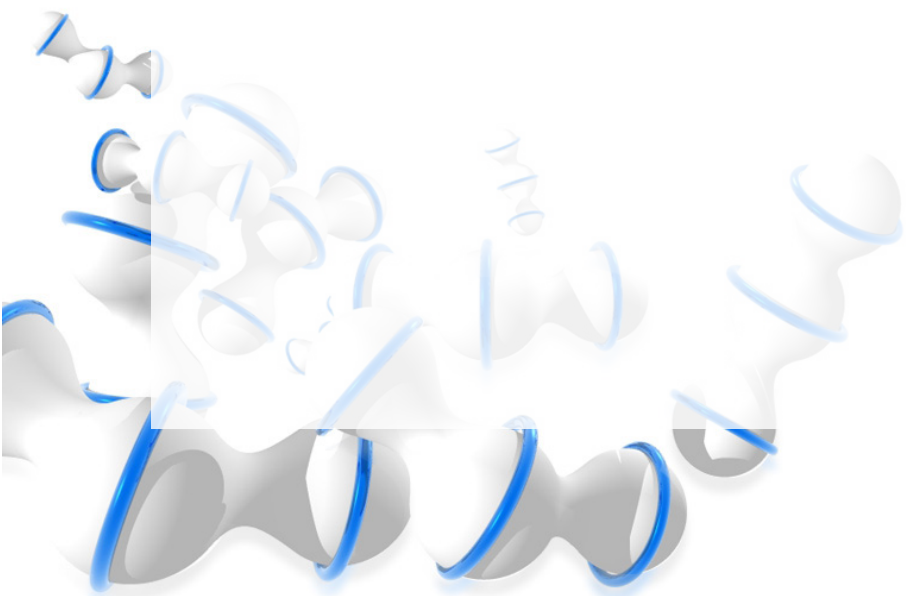
Techniken: Array-basierte Sprachkonzepte

- Verfeinerte Variante der komplett übersetzten Webseiten:
 - Nur Schnipsel übersetzen (engl. „*snippets*“)
- Datei mit Array
 - Schlüssel: Platzhalter für Snippets
 - Werte: Ausdrücke in der jeweiligen Sprache



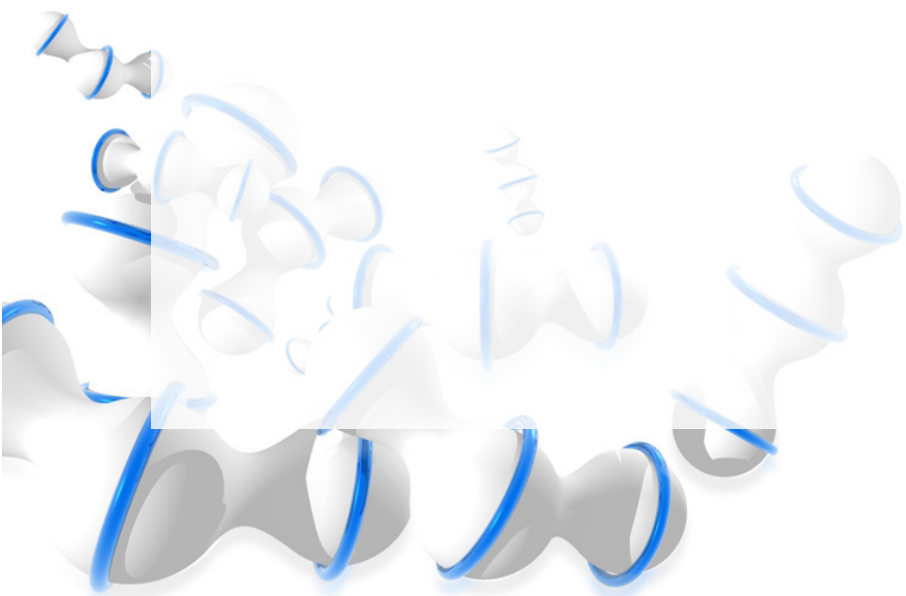
Techniken: Array-basierte Sprachkonzepte

- Template mit Platzhaltern laden
- Sprachlich passendes Array laden und Platzhalter ersetzen
- Übrige in Standardsprache ersetzen
- Seite an Benutzer ausliefern



Techniken: Array-basierte Sprachkonzepte

- Vorteil:
 - losere Kopplung von Formatierung und Inhalt als bei der Verwendung von kompletten internationalisierten HTML-Seiten
 - -> verbessertes Konzept!



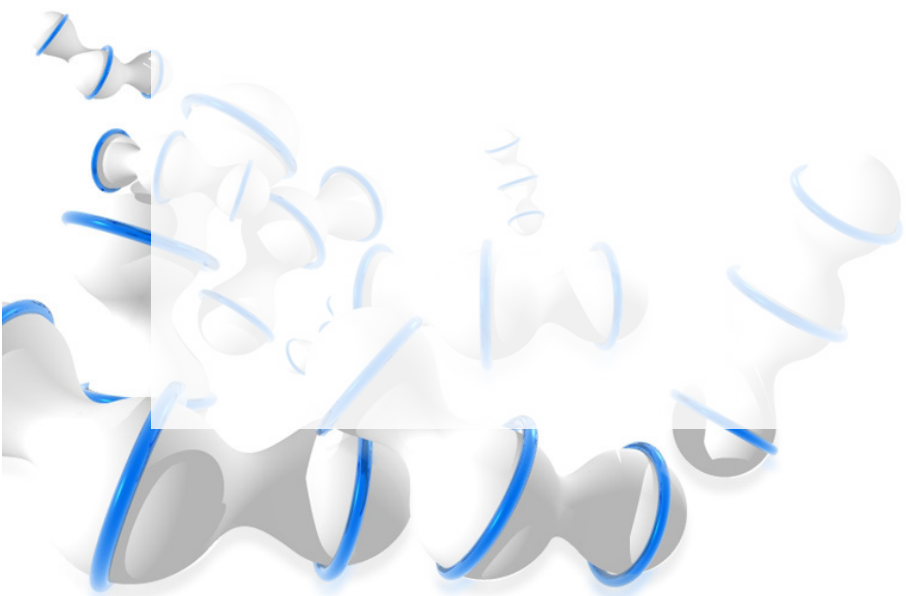
Techniken: Array-basierte Sprachkonzepte

- Nachteile:
 - Schwer wartbar: Außenstehende müssen wissen wie Platzhalter heißen und wofür sie stehen
 - Unklare Trennung von Formatierung und Inhalt
 - Lücken bei fehlenden Übersetzungsschnipseln



Techniken: Gettext

- GNU Gettext = GNU-Internationalisierungs-bibliothek
- Für Entwicklung mehrsprachiger Programme
- Bei OpenSource-Gemeinde sehr beliebt



Techniken: Gettext

- Erzeugt .po-Datei (= '*portable object*'), enthält:
 - Reine, übersetzte Information
 - Daten zur übersetzten Textstelle
 - Meta-Informationen
 - Einfügungsstellen für dynamische Textelemente (z.B. aktueller Wochentag)



Techniken: Gettext

- Portable Object-Datei:
- Zur Auslieferung: Übersetzung in .mo-Dateien (= '*machine object*')
 - Enthält Originaltext in kleinen, zu übersetzenden Abschnitten
 - Hier kein Problem: Nicht vorhandene Übersetzungsschnipsel – Originaltext anzeigbar!



Techniken: Gettext

- Beispiel: test.po

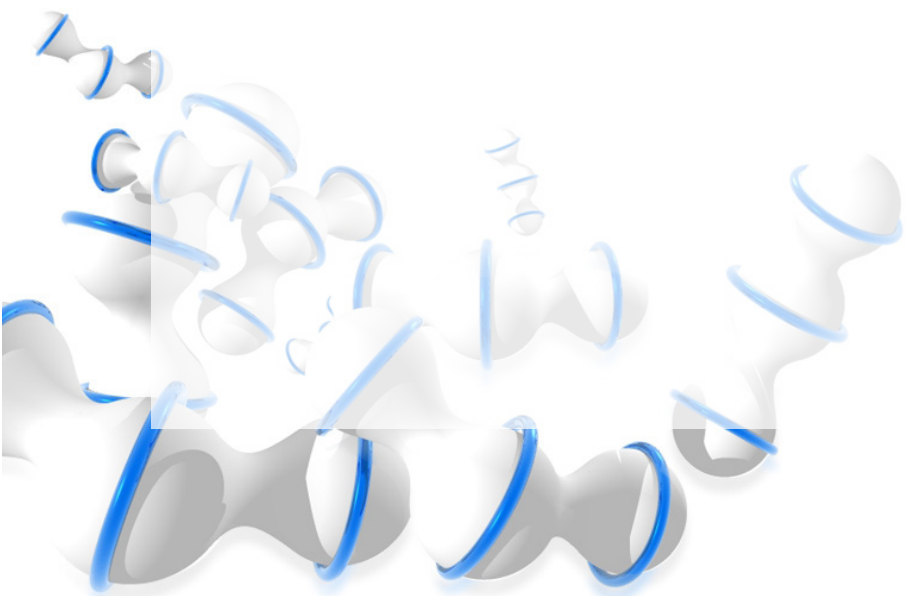
```
#TITEL: IMPRESSUM
#COPYRIGHT (C) 2007
#STEFFEN ELLER
#this file is distributed under the same license as the ... package
#FIRST AUTHOR: STEFFEN ELLER<STEFFEN.ELLER@MNI.FH-GIESSEN.DE>, 2006
#
#, fuzzy
msgid""
msgstr""
"Project-Id-Version:1.0\n"
"Report-Msgid-Bugs-To: \n"
"POT-Creation-Date 2006-08-07 12:45+0200\n"
"POT-Revision-Date 2006-08-08 15:22+0200\n"
"Last-Translator:STEFFEN ELLER<STEFFEN.ELLER@MNI.FH-GIESSEN.DE>\n"
"Language-Team:GERMAN<germany@languages.com>\n"
"MIME-Version:1.0\n"
"Content-Type:text/plain;charset=ISO-8859-1\n"
"Content-Transfer-Encoding:8bit\n"
```

Techniken: Gettext

- Beispiel: test.po

```
#: impressum.php:36  
msgid "My name is Steffen.\n"  
msgstr "Mein Name ist Steffen.\n"
```

```
#: impressum.php:42  
msgid "How are you?\n"  
msgstr "Wie geht es dir?\n"
```



Techniken: Gettext

- Gettext in php:
 - Sprache festlegen und mittels `setlocale()` ; ins System exportieren
 - Gettext-Domain (Referenz) auf `.mo`-Dateien
 - Strings mit `_(„Mein String“)` ; oder `gettext(„Mein String“)` ; aufrufen



Techniken: Gettext

- Beispiel php:

```
<?php
// Sprache festlegen (hier: auf Deutsch)
$language = "de_DE";
putenv("LANG=$language");
setlocale(LC_ALL, $language);

// Ort der Uebersetzungstabellen angeben
$domain="test";
bindtextdomain($domain, "./locale");
textdomain($domain);

// Uebersetzung wird nun aus ./locale/de_DE/LC_MESSAGES/test.mo geholt

// Zu Uebersetzender Teil
echo ("My name is Steffen.\n");
echo "<br />";
echo gettext("How are you?\n");
?>
```

- Ausgabe:

```
Mein Name ist Steffen.\n
<br />
Wie geht es dir?\n
```

Techniken: Gettext

- Vorteile:
 - Mächtig
 - Kompakt
 - Verständlich
- Nachteile:
 - setlocale()-Nebenwirkungen (Nebenläufigkeit)
 - Berechnungsaufwand

Techniken: Stringlisten

- Ähnlich array-basiertem Sprachkonzept und Gettext
- Datei im ini-Format (in jeder Zeile Bezeichner=übersetzter Inhalt)
- Beispiel: demo.list



```
my.name.is=Mein Name ist
```

```
date.time=Die aktuelle Uhrzeit ist %s, das aktuelle Datum lautet %s
```

Techniken: Stringlisten

- php-Implementierung: Stream-Wrapper Klasse

```
<?php
```

```
class dtdStream {
```

```
    private $uri;  
    private $fp;
```

```
    const BASEDIR='/webpace/i18n';  
    public static $lang;
```

```
    // nicht implementierte Funktionalitaet abfangen
```

```
    public function __call($name,$args) {  
        trigger_error("Funktion '$name' nicht implementiert.");  
    }
```

Techniken: Stringlisten

```
// Wrapper Registrierung
static function register($n='locale') {
    stream_wrapper_register($n, "dtdStream");
    return true;
}

// Pfad Uebersetzung
private function translate($path) {
    // Regex zum URI RFC #2396 - Regex result array:

    $tmp=array();

    preg_match("=^(([^:/?#]+):)?(//([^/?#]*))?([^?#]*)(\\?([^#]*)?)?(#(.*))?"
    "=", $path, $tmp);

    // realen Dateisystempfad ablegen
    return self::BASEDIR.'/'.$self::$lang.'/'.$tmp[4].$tmp[5];
}
```

Techniken: Stringlisten

```
public function stream_open($path,$mode,$options,&$opened_path) {  
    // realen Dateisystempfad ablegen  
    $this->uri=$this->translate($path);  
  
    // Gibt's die Datei?  
    if (!file_exists($this->uri)) { return false; }  
  
    // Datei Oeffnen  
    $this->fp=fopen($this->uri, $mode, $options);  
    if (!$this->fp) { // irgendwas ging schief...  
        return false;  
    }  
    return true;  
}  
  
public function stream_close() {  
    fclose($this->fp);  
    $this->fp=0;  
    return;  
}  
  
public function stream_read($count) {  
    return fread($this->fp,$count);  
}
```


Techniken: Stringlisten

```
public function stream_eof() {    return feof($this->fp); }

public function stream_tell() {    return ftell($this->fp); }

public function stream_seek($offset,$whence){
    return fseek($this->fp,$offset,$whence);
}

public function stream_stat() {    return fstat($this->fp); }

public function url_stat($path,$flags) {
    return stat($this->translate($path));
}

public function stream_flush() {
    // readonly, also nichts zu tun
    return true;
}

} // dtdStream

?>
```

Techniken: Stringlisten

- php-Aufruf: Stream-Wrapper Klasse

```
<?php
```

```
// Klasse streamwrapper einbinden, um Methoden verwenden zu koennen  
require('./streamwrapper.inc.php');
```

```
// Wrapper registrieren  
dtdStream::register();
```

```
// Sprache auf Deutsch einstellen  
dtdStream::$lang='de_DE';
```

```
/* Klasse stringbundle einbinden:  
 * stellt die Stringlisten-Paare (Bezeichner=Sprachspezifischer  
Kontent)  
 * zur Verfuegung  
 */  
require('./stringbundle.php');
```

Techniken: Stringlisten

```
// Instanz anlegen
$strBndl=stringBundle::getInstance();

// Stringlisten-Datei laden
$strBndl->load('locale://strings/demo.list');

// String holen, je nach voreingestellter $lang
echo $strBndl->getString('my.name.is');
echo "<br/>";

// Formatierten String holen, je nach voreingestellter $lang
echo $strBndl-
    >getFormattedString('date.time',strftime('%X'),strftime('%x'));
```

?>

Techniken: Stringlisten

- Vorteile:
 - von Programmiersprache unabhängig
 - einfache Syntax
- Nachteile:
 - fehlende Mehrzeiligkeit



Techniken: Entities

- Gettext-, Array- & Stringlisten-Varianten speziell für dynamischen Inhalt
- HTML-Seiten sind aber oft auch statisch
- Idee: XHTML-Variante
 - Dadurch direkte XML-Konformität
- Realisierung: Selbstdefinierte Entitäten
- Bsp.: Mozilla-Produkte, PHP-Dokumentation

Techniken: Entities

- Implementierung

```
<?php
```

```
// streamwrapper laden  
require('./streamwrapper.inc.php');
```

```
// Wrapper registrieren  
dtdStream::register();
```

```
// Sprache auf Deutsch einstellen  
dtdStream::$lang='de';
```

```
// DomDocument-Instanz erzeugen  
$src=new DomDocument;
```

```
// Entity ersetzen aktivieren  
$src->substituteEntities = true;  
$src->resolveExternals   = true;
```

```
// HTML Quelldatei laden  
$src->load('test.html');
```

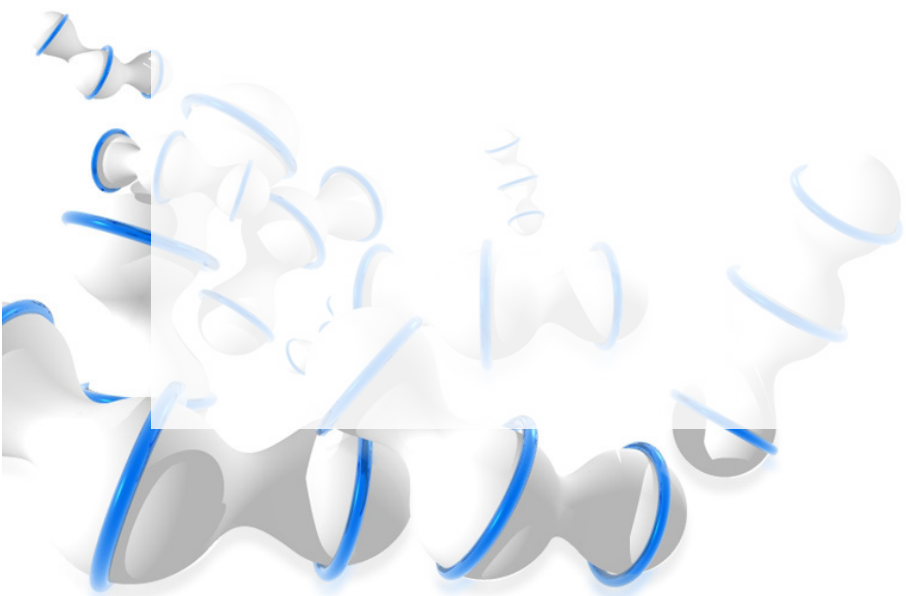

Techniken: Entities

```
// leeres xhtml Dokument erzeugen
$doctype = DOMImplementation::createDocumentType("html",
    "-//W3C//DTD XHTML 1.0 Transitional//EN",
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd");
$html = DOMImplementation::createDocument(null, null, $doctype);

// Quell-Daten einbinden
$html->appendChild($html->importNode($src->documentElement,true));

// Ergebnis ausgeben...
echo $html->saveHtml();

?>
```



Techniken: Entities

- Template:

```
<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE html [
  <!ENTITY % localeDTD SYSTEM "locale://dtd/test.dtd">
  %localeDTD;
]>
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>&test.title;</title>
  </head>
  <body>
    <p>&test.abschnitt;</p>
  </body>
</html>
```

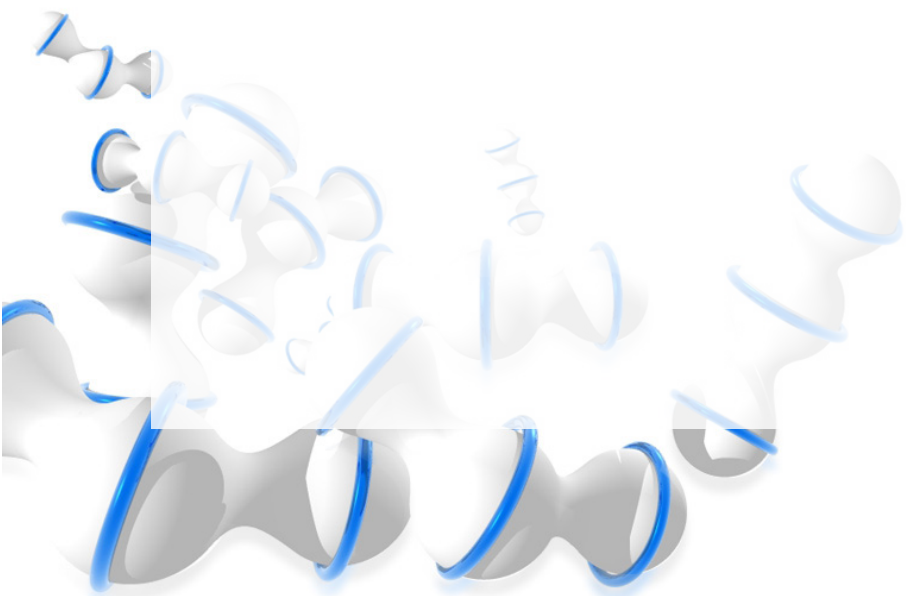
Techniken: Entities

- Vorteile:
 - klare Trennung von Layout und Inhalt
 - Performance
 - einfache Validation
- Nachteile:
 - nicht übersetzte Entities



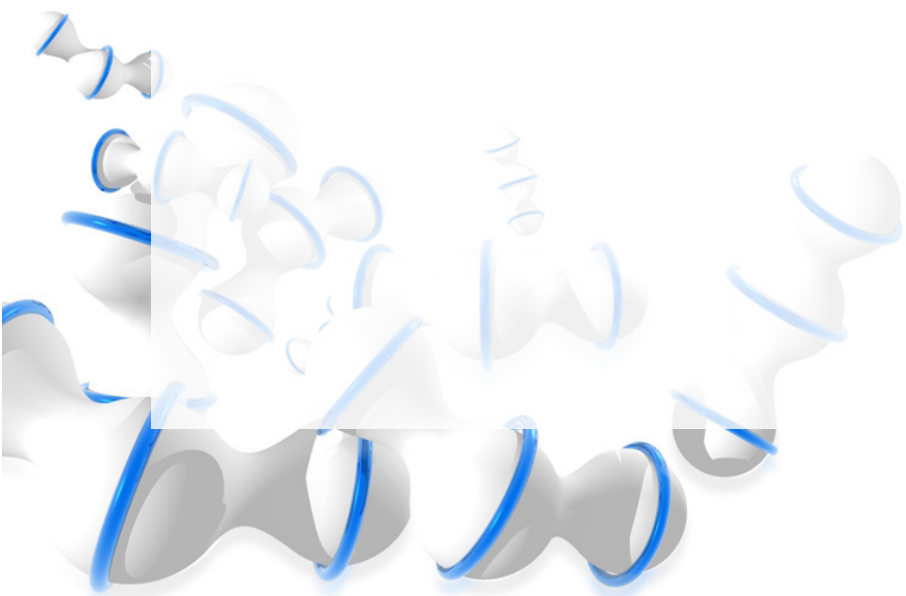
Fazit

- Immer: Abschätzung Aufwand vs. Nutzen
- Strukturierte Internationalisierung bei größeren Projekt unabkömmlich
- Erfolg bringt Kombination verschiedener Techniken



Ende

Vielen Dank für eure Aufmerksamkeit!



Fragen & Diskussion

